Mise en pratique de LSPI pour la commande linéaire quadratique adaptative d'une surface de manipulation à coussin d'air actif.

Guillaume J. Laurent

Mise en pratique de LSPI pour la commande linéaire quadratique adaptative d'une surface de manipulation à coussin d'air actif

Guillaume J. Laurent

Institut FEMTO-ST, UFC-ENSMM-UTBM-CNRS
Département Automatique et Systèmes Micro-Mécatroniques
24 rue A. Savary, 25000 Besançon, France
guillaume.laurent@ens2m.fr

Résumé : Cet article présente l'application de l'algorithme LSPI de Lagoudakis & Parr (2003) à la commande d'un système linéaire avec coût quadratique selon le protocole initialement proposé par Bradtke (1993). Le dispositif contrôlé est une surface active capable de mouvoir un objet sur un coussin d'air et dont la dynamique varie fortement en fonction de l'objet utilisé. La méthode d'apprentissage est validée en simulation avant d'être appliquée au système réel. Les résultats expérimentaux mettent en évidence la nécessité de formater les commandes générées par l'algorithme. Ce formatage a pour objectif d'éviter la génération de commandes irréalisables qui introduisent un biais dans la mise à jour de la fonction de valeur. L'apprentissage converge alors vers la même solution que la commande linéaire quadratique.

Mots-clés : apprentissage par renforcement, LSTD, LSTDQ, LSPI, commande linéaire quadratique, LQR.

1 Introduction

Les récents travaux en apprentissage par renforcement permettent d'envisager l'adaptation en ligne de la commande de systèmes réels. Nous nous intéressons dans cet article aux systèmes linéaires qui constituent une grande part des systèmes commandés. Depuis longtemps, la commande linaire quadratique permet de calculer une loi de commande (en d'autres termes une politique d'action) optimale pour un processus linéaire. Si le modèle du processus n'est pas le reflet exact de la réalité ou si certains paramètres évoluent, la loi de commande n'est plus optimale et les performances sont dégradées, le système bouclé pouvant même devenir instable. L'apprentissage par renforcement est un paradigme qui a pour objectif d'optimiser une loi de commande à l'aide de données expérimentales. Cette méthode permet donc, en théorie, d'adapter la loi de commande à la véritable dynamique du processus. La technique a montré son efficacité sur de nombreux systèmes complexes simulés mais les résultats expérimentaux restent encore peu nombreux.

Cet article propose d'appliquer l'algorithme LSPI [1] de Lagoudakis & Parr (2003) à l'adaptation en ligne d'une loi de contrôle linéaire quadratique selon le protocole initialement proposé par Bradtke (1993). Le système commandé est une surface de manipulation sans contact capable de déplacer un objet sur un coussin d'air à l'aide de jets d'airs. La problématique de sa commande est que la dynamique du mouvement de l'objet dépend fortement de sa géométrie et de sa masse. Il serait donc intéressant que le contrôleur adapte automatiquement ses commandes quand un nouvel objet est posé sur la surface.

La première partie de l'article présente succinctement la commande linéaire quadratique. La seconde est consacrée à l'algorithme LSPI dans le cadre de la commande de systèmes linéaires avec coût quadratique. La troisième détaille le fonctionnement du dispositif expérimental et l'objectif du contrôle. Enfin, les deux dernières parties présentent les résultats obtenus en simulation et avec le processus réel.

1. Least-Squares Policy Iteration

2 Commande linéaire quadratique

La représentation d'état en temps discret d'un processus linéaire par rapport à ses entrées est un système d'équations de la forme :

$$\begin{cases} X_{k+1} & = AX_k + BU_k \\ Y_k & = CX_k + DU_k \end{cases} \tag{1}$$

où :
- X_k est le vecteur d'état qui représente les n variables d'état du processus à l'instant k ;
- U_k est le vecteur de commande qui représente les m commandes appliquées au processus à l'instant k ;
- Y_k est le vecteur de sortie qui représente les p observations effectuées sur le processus à l'instant k ;
- A est la matrice de dynamique de dimension $n \times n$;
- B est la matrice de commande de dimension $n \times m$;
- C est la matrice d'observation de dimension $p \times n$;
- D est la matrice d'action directe de dimension $p \times m$.

Ce système constitue naturellement un processus décisionnel de Markov partiellement observable.

L'objectif de la régulation linéaire quadratique (LQ) est de ramener l'état X du processus à un état d'équilibre (ici 0 pour simplifier) à partir d'un état initial quelconque X_0. La loi de commande (ou politique) est un retour d'état de la forme :

$$U_k = \pi(X_k) = K.X_k \tag{2}$$

où K est une matrice dimension $m \times n$.

La particularité de la commande LQ est de déterminer un correcteur K^* minimisant une fonction de coût quadratique (ou fonction de valeur) de la forme :

$$V^K(X) = \sum_{k=0}^{\infty} r(X_k, U_k) = \sum_{k=0}^{\infty} X_k' E X_k + U_k' F U_k, \quad X_0 = X \tag{3}$$

où E et F sont des matrices symétriques définies positives de dimensions respectives $n \times n$ et $m \times m$.

Dans le cas où les matrices A et B sont connues et si la paire (A, B) est commandable, on peut déterminer K^* en résolvant une équation de Riccati. On a :

$$K^* = -(B'PB + F)^{-1} B'PA \tag{4}$$

où P est la matrice de coût solution de l'équation de Riccati suivante :

$$E + A'PA - A'PB(B'PB + F)^{-1} B'PA = P \tag{5}$$

On a alors :

$$V^*(X) = X'PX \tag{6}$$

Pour plus de détails, on pourra consulter par exemple (Bertsekas, 2005).

Dans le cas où les matrices A et B sont inconnues, une solution envisageable est d'optimiser K par apprentissage par renforcement à l'aide des données expérimentales.

3 LSPI pour processus linéaires avec coût quadratique

3.1 Fonction de valeur d'action

Quand le modèle d'évolution du processus est inconnu, il est nécessaire de minimiser une fonction de valeur de l'action afin de pouvoir en extraire une politique gloutonne. La fonction de valeur d'action pour le critère $gamma$-pondéré est définie par la fonction Q^π telle que :

$$Q^\pi(X, U) = \mathbb{E}\left[\sum_{k=0}^{\infty} \gamma^k r(X_k, U_k) | X_0 = X, U_0 = U, \pi\right] \tag{7}$$

γ est le coefficient d'actualisation. π est la politique suivie, en d'autres termes la loi de commande appliquée au processus.

Dans le cas d'un système linéaire avec une fonction de coût quadratique, la fonction de valeur d'action est de la forme (Bradtke, 1993) :

$$Q^K(X_k, U_k) = \begin{bmatrix} X_k & U_k \end{bmatrix} \begin{bmatrix} E + \gamma A'PA & \gamma A'PB \\ \gamma B'PA & F + \gamma B'PB \end{bmatrix} \begin{bmatrix} X_k \\ U_k \end{bmatrix} \tag{8}$$

$$\tag{9}$$

où P est la matrice de coût pour le correcteur K.

On définit alors :

$$H = \begin{bmatrix} E + \gamma A'PA & \gamma A'PB \\ \gamma B'PA & F + \gamma B'PB \end{bmatrix} = \begin{bmatrix} H_{11} & H_{12} \\ H'_{12} & H_{22} \end{bmatrix} \tag{10}$$

H est une matrice symétrique définie positive de dimensions $(n+m) \times (n+m)$. La fonction de valeur est donc représentée par $(n+m)(n+m+1)/2$ coefficients.

L'algorithme LSPI présenté plus loin suppose que la fonction de valeur soit définie par un modèle linéaire par rapport à ses paramètres :

$$Q_t(X, U) = \sum_{i=1}^{n} \phi_i(X, U)\theta_{t,i} = \phi'(X, U)\theta_t \tag{11}$$

La fonction de valeur d'action (8) est bien linéaire par rapport à ses paramètres, en effet si on pose :

$$\phi(X, U) = \begin{bmatrix} x_1^2 & 2x_1x_2 & 2x_1x_3 & \cdots & 2x_1u_m & x_2^2 & 2x_2x_3 & \cdots & u_m^2 \end{bmatrix}' \tag{12}$$

et :

$$H = \begin{bmatrix} \theta_1 & \theta_2 & \theta_3 & \cdots & \theta_{n+m} \\ \theta_2 & \theta_{n+m+1} & \theta_{n+m+2} & \cdots & \theta_{2n+2m-1} \\ \theta_3 & \theta_{n+m+2} & \theta_{2n+2m} & & \vdots \\ \vdots & \vdots & & \ddots & \vdots \\ \theta_{n+m} & \theta_{2n+2m-1} & \cdots & \cdots & \theta_{(n+m)(n+m+1)/2} \end{bmatrix} \tag{13}$$

On a bien :

$$Q_t(X, U) = \begin{bmatrix} X & U \end{bmatrix} H_t \begin{bmatrix} X \\ U \end{bmatrix} = \phi'(X, U)\theta_t \tag{14}$$

3.2 LSTD et LSTDQ

Comme tous les algorithmes d'apprentissage par renforcement, les algorithmes de la famille du LSTD [2] cherchent à minimiser la fonction de valeur à l'aide des informations courantes $(X_k, U_k, r_{k+1}, X_{k+1})$. Introduit par Bradtke & Barto (1996), l'algorithme LSTD permet de minimiser la fonction de valeur V de façon particulièrement efficace. Une « variante », nommée LSTDQ, permettant d'optimiser la fonction de valeur d'action Q a été proposée par Lagoudakis & Parr (2003). Une version avec traces d'éligibilité a également été proposée par Boyan (2002) ainsi que par Peters & Schaal (2008).

Le LSTDQ cherche à minimiser la fonction de valeur d'action en comparant à chaque transition l'estimation du coût $\hat{r}(X_k, U_k) = [\phi(X_k, U_k) - \gamma\phi(X_{k+1}, U_{k+1})]\theta_t$ avec le coût reçu $r(X_k, U_k)$. Le principe du LSTDQ est d'utiliser $\phi(X_k, U_k)$ comme variable instrumentale. En effet, $\phi(X_k, U_k)$ est corrélé avec

2. Least-Squares Temporal Difference

$\hat{r}(X_k, U_k)$ et n'est pas corrélé avec l'erreur de mesure $\gamma \sum_{Z \in \mathcal{X}} T(X_k, U_k, Z)\phi(X', U') - \gamma\phi(X_{k+1}, U_{k+1})$ (où $T(X_k, U_k, Z)$ est la fonction de transition du processus). On a alors :

$$\theta_{t+1} = \left[\frac{1}{M} \sum_{k=0}^{M-1} \phi(X_k, U_k)[\phi(X_k, U_k) - \gamma\phi(X_{k+1}, \pi_t(X_{k+1}))]' \right]^{-1} \left[\frac{1}{M} \sum_{k=0}^{M-1} \phi(X_k, U_k) r(X_k, U_k) \right] \tag{15}$$

On peut déduire une formulation récursive de cette équation qui évite l'inversion de matrice (Bradtke & Barto, 1996; Lagoudakis & Parr, 2003; Geramifard *et al.*, 2006).

3.3 LSPI pour processus linéaires avec coût quadratique

Dans cet article, nous avons mis en pratique un schéma d'itération de la politique fondé sur LSTDQ, appelé LSPI [3] et qui a été proposé par Bradtke (1993) dans le cadre des systèmes linéaires avec coût quadratique puis généralisé par Lagoudakis & Parr (2003) au cas des fonctions de valeurs linéaires par rapport à leurs paramètres. Cette méthode permet d'améliorer la politique après chaque essai.

En dérivant l'équation (8), il est aisé de montrer qu'une politique gloutonne [4] sur Q est définie par :

$$\forall X, \quad \pi_{t+1}(X) = \arg\min_V Q_{t+1}(X, V) = \arg\min_V \phi'(X, V)\theta_{t+1} = -H_{22}^{-1} H_{21} X \tag{16}$$

donc le nouveau correcteur est :

$$K_{t+1} = -H_{22}^{-1} H_{21} \tag{17}$$

L'algorithme 20 résume la démarche mise en œuvre. Le LSPI pour processus linéaires avec coût quadratique présente les avantages suivants :

- le nombre de paramètres de la fonction de valeur est $(n + m)(n + m + 1)/2$ où n est la dimension de l'espace d'état et m celle de l'espace de commande ; Cette représentation est très compacte par rapport à des fonctions d'approximation utilisant des fonctions radiales ;
- Si le couple (A, B) est commandable, si K_0 est stabilisant et si la commande appliquée au processus comporte une part d'exploration suffisante, il existe un nombre fini d'échantillons M tel que la politique K_t converge vers la politique optimale (Bradtke, 1994) (Bradtke *et al.*, 1994) ;
- le calcul de la politique gloutonne est analytique et ne nécessite donc aucune méthode d'optimisation spécifique. De plus, ce calcul est effectué hors ligne.

4 Surface de manipulation à coussin d'air actif

4.1 Principe de fonctionnement

La surface de manipulation à coussin d'air actif est un plan de 120×120 mm^2 sur lequel un objet peut se déplacer en lévitation aérodynamique (cf. figure 1a). La surface est percée de petits trous ayant des rôles différents. Un trou sur deux permet de générer le coussin d'air assurant la lévitation de l'objet. Ces orifices « de lévitation » sont alimentés par une seule entrée d'air à une pression de 10 kPa.

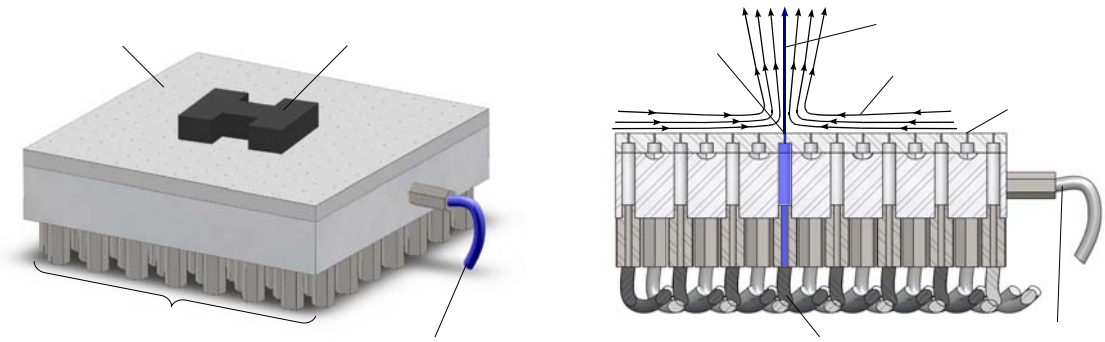
Les autres orifices, dits « de traction », permettent de générer de puissants jets d'air verticaux qui entraînent indirectement un mouvement de l'objet. Chaque jet d'air est alimenté par une canalisation individuelle connectée à une électrovanne. Les électrovannes permettent ainsi de piloter la présence ou non de jets d'air de traction. La pression d'alimentation des électrovannes est de 500 kPa.

L'objet est ainsi déplacé par l'action indirecte des jets d'airs de traction. Quand une électrovanne s'ouvre, un puissant jet d'air vertical est généré en un point de la surface. Ce jet d'air entraîne l'air ambiant et crée un flux d'air induit qui tire l'objet vers l'orifice (cf. figure 1b). Cet effet est un phénomène contre-intuitif de la mécanique des fluides comme la célèbre lévitation de Bernoulli (Waltham *et al.*, 2003).

3. Least-Squares Policy Iteration
4. Une politique gloutonne sur Q consiste à choisir en tout état une commande qui minimise la fonction de valeur.

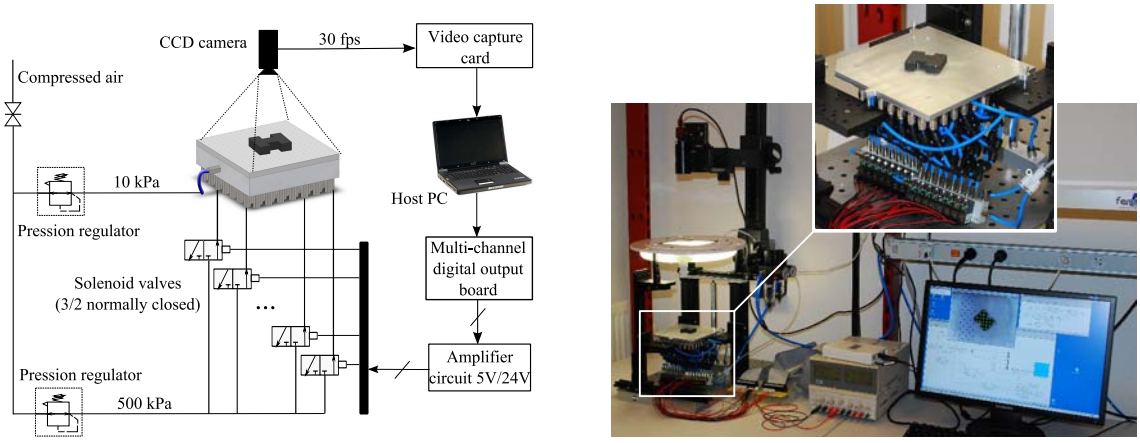
Algorithme 1 : LSPI pour processus linéaires avec coût quadratique (Bradtke, 1993).

début
  Initialiser le correcteur initial K_0 (doit être stabilisant)
  $t \leftarrow 0$
  répéter
    $\Phi \leftarrow 0$
    $\rho \leftarrow 0$
    Obtenir l'état initial X_0 du processus
    pour k *de* 0 *à* $M-1$ **faire**
      $U_k \leftarrow K_t X_k + e_k$ (où e_k est un signal d'exploration); `(partie en ligne)`
      Appliquer U_k au processus et attendre X_{k+1}
      $U^{on} \leftarrow K_t X_{k+1}$
      $\Phi \leftarrow \Phi + \phi(X_k, U_k)[\phi(X_k, U_k) - \gamma\phi(X_{k+1}, U^{on})]$
      $\rho \leftarrow \rho + \phi(X_k, U_k) r(X_k, U_k)$
    fin
    $\theta_{t+1} \leftarrow \Phi^{-1}\rho$; `(partie hors ligne)`
    Trouver les coefficients de H correspondants aux paramètres θ_{t+1}
    $K_{t+1} \leftarrow -H_{22}^{-1} H_{21}$
    $t \leftarrow t+1$
  jusqu'à $\theta_t \approx \theta_{t-1}$
fin



(a) Vue en perspective (b) Vue en coupe

FIGURE 1 – Surface de manipulation à coussin d'air actif.



(a) Schéma du dispositif

(b) Aperçu du dispositif

FIGURE 2 – Dispositif expérimental.

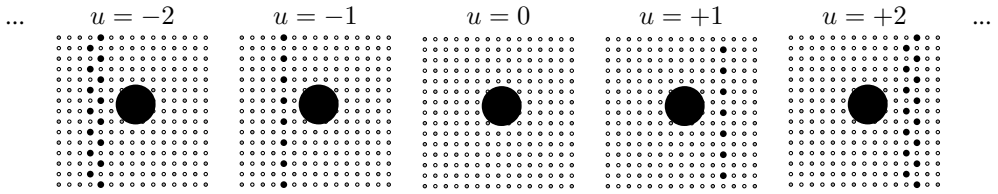


FIGURE 3 – La commande u est un entier représentant le nombre de colonnes ouvertes. Le signe de u désigne le côté de l'ouverture. Les cercles gris correspondent à des orifices fermés, les petits disques noirs à des orifices ouverts. Le grand disque noir représente l'objet.

4.2 Dispositif expérimental

Le dispositif expérimental est constitué d'une alimentation en air comprimé, de deux régulateurs de pression (un pour le coussin d'air et un pour les jets d'air de traction), d'un ensemble d'électrovannes, d'une interface de commande connectée à un ordinateur et d'un système d'acquisition vidéo. La figure 2 montre un aperçu du dispositif.

La position de l'objet est mesurée à l'aide de la caméra et d'un logiciel de vision (bibliothèque cvLink fondée sur OpenCV) à la fréquence de 30 Hz. Dès que la nouvelle position est connue, le contrôleur doit décider d'une nouvelle commande qu'il envoie aux électrovannes. L'objectif à long terme est de contrôler les trois degrés de liberté de l'objet (deux translations et une rotation) par une combinaison adaptée de l'ouverture des électrovannes.

4.3 Régulation de position 1D

Dans cet article, nous nous sommes focalisés sur l'asservissement de la position de l'objet dans une seule direction (axe est-ouest). L'objet utilisé est une pièce cylindrique en aluminium de 30 mm de diamètre, 15 mm de haut et de 29 g.

Pour réaliser un mouvement dans la direction est-ouest, tous les orifices de traction d'une même colonne sont reliés à une même électrovanne. Une électrovanne pilote donc une colonne de jets d'air. Le signal de commande peut se résumer à un nombre entier donnant le nombre de colonnes à ouvrir d'un côté ou de l'autre de l'objet (cf. figure 3).

La figure 4 montre un exemple de trajectoire obtenue quand une commande constante $u = 2$ est appliquée au système. Cet essai montre que la réponse du système est un second ordre avec intégrateur (et que le système est simplement stable). Le problème est que cette réponse dépend de la géométrie et de la masse de l'objet utilisé. Ainsi, si on calcule un correcteur optimal K^* pour un certain objet, le contrôle n'est pas

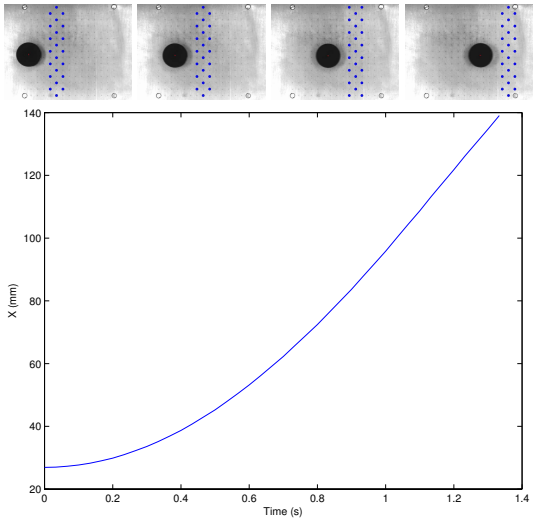


FIGURE 4 – Séquence d'images et position de l'objet en fonction du temps pour une commande constante $u = 2$ (courbe expérimentale).

adapté à un autre objet. Pour cette raison, nous souhaitons optimiser le correcteur K en ligne de façon que la loi de commande s'adapte à chaque objet.

Afin de comparer les différentes approches, nous avons choisi de tester une tâche de régulation à 0. A l'instant $t = 0$ s, l'objet est placé à 20 mm de l'origine sans vitesse initiale (l'origine est située au milieu de la surface). Le correcteur doit ramener l'objet à la position 0 en minimisant le critère (3) défini avec le coût immédiat suivant :

$$r(x, u) = x^2 + 10u^2 \tag{18}$$

où x est la position de l'objet et u la commande appliquée. Cette fonction de coût a été choisie pour éviter une saturation trop importante des commandes. Un essai dure 10 s à raison de 30 échantillonnages par seconde, soit 300 pas de calculs ($M = 300$).

5 Résultats de simulation

5.1 Identification

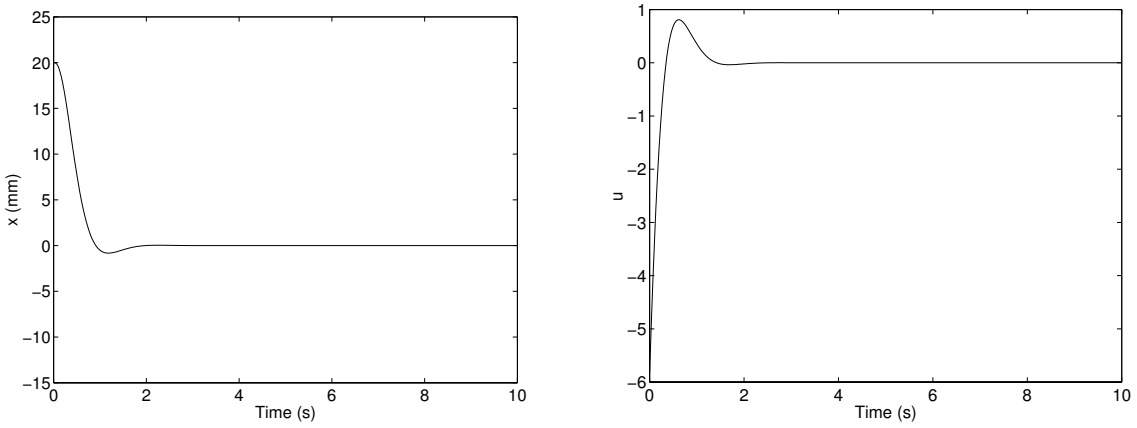
Avant de tester l'algorithme sur le système réel, nous avons réalisé des essais en simulation. Dans ce but, nous avons procédé à une identification paramétrique. Le système est d'ordre trois. Son entrée U est la commande u envoyée aux électrovannes. Sa sortie Y est la position de l'objet x (abscisse). La fréquence d'échantillonnage est donnée par la cadence de l'acquisition vidéo (30 Hz). L'identification donne le modèle suivant :

$$\begin{cases} X_{k+1} = \begin{bmatrix} 1.0000 & 0 & 0 \\ 0 & 0.9578 & 0 \\ 0 & 0 & 0.6756 \end{bmatrix} X_k + \begin{bmatrix} -0.0622 \\ 0.0621 \\ 0.0598 \end{bmatrix} u_k \\ x_k = \begin{bmatrix} -22.3632 & -24.6381 & 2.3784 \end{bmatrix} X_k \end{cases} \tag{19}$$

5.2 Commande linaire quadratique

Nous avons alors calculé un correcteur optimal K^* pour la fonction de coût (18). Ce coût correspond aux matrices $E = C'C$ et $F = 10$ puisque $x = CX$ et $u = U$. En résolvant l'équation de Riccati, on trouve :

$$K^* = \begin{bmatrix} -6.5520 & -4.1332 & 0.0218 \end{bmatrix} \tag{20}$$



(a) Position de l'objet en fonction du temps

(b) Commande en fonction du temps

FIGURE 5 – Résultats de simulation avec le correcteur LQ.

La figure 5 montre la position de l'objet et la commande en fonction du temps avec le correcteur K^* appliqué à la simulation du système. Le correcteur est bien stabilisant, le dépassement est relativement faible et l'oscillation est vite amortie. Le système étant intégrateur, l'erreur statique est naturellement nulle. Le critère est égal à $V^*(X_0) = 5.3135\text{e}3$ avec $X_0 = \begin{bmatrix}-0.8943 & 0.0000 & 0.0000\end{bmatrix}'$.

5.3 Apprentissage *ex nihilo*

Nous avons appliqué l'algorithme 20 à la simulation du système. Pour assurer sa convergence, l'algorithme doit être initialisé avec un correcteur K_0 stabilisant le système. Le système étant simplement stable (intégrateur), nous pouvons choisir $K_0 = \begin{bmatrix}0 & 0 & 0\end{bmatrix}'$.

Le signal d'exploration e_k utilisé est un bruit gaussien de moyenne nulle et de variance 5 ; γ est fixé à 1 (l'horizon temporel est fini). Pour évaluer le critère $V(X_0)$, on utilise $e_k = 0$.

La figure 6a montre l'évolution du critère $V(X_0)$ en fonction du temps. La politique optimale est obtenue en quelques itérations. La courbe de la figure 6b représente la position de l'objet en fonction du temps après l'apprentissage. La trajectoire finale de l'objet est identique à celle de la figure 5.

L'intérêt de cette méthode par rapport à la résolution de l'équation de Riccati est qu'elle ne nécessite pas la connaissance des matrices A, B, C et D mais simplement celles de l'ordre du système et de la fonction de coût.

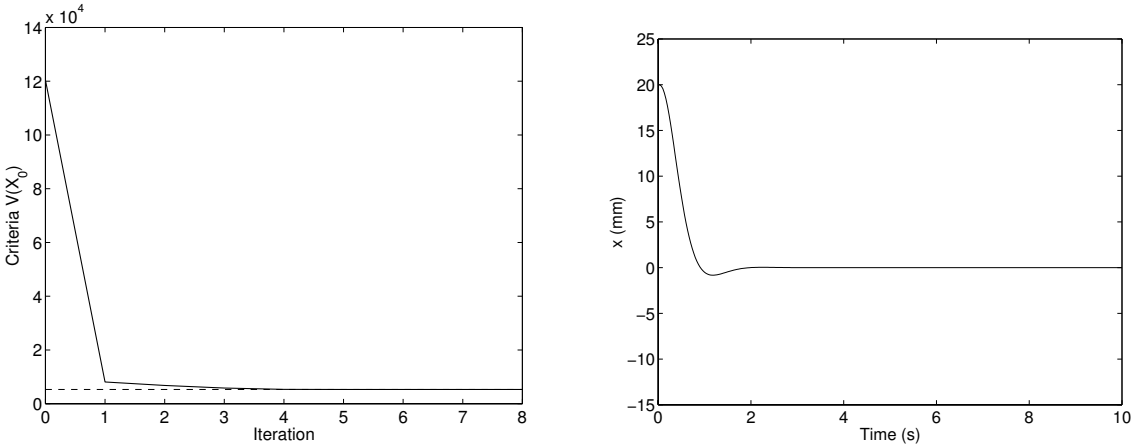
5.4 Discussion

Avant d'aboutir à ces résultats, nous avons rencontré de nombreux problèmes de stabilité numérique. En effet, la matrice Φ n'était pas toujours inversible et particulièrement mal conditionnée. Pour y remédier, nous avons pris les précautions suivantes :

- la représentation d'état (base de X) doit être choisie pour que la matrice A soit bien conditionnée ;
- le bruit d'exploration influe sur le conditionnement de Φ ; d'une part, l'exploration doit être suffisamment forte pour maintenir un bon conditionnement ; d'autre part, l'exploration ne doit pas trop déstabiliser le système.

Malgré ces précautions, la matrice Φ reste moyennement conditionnée lors des simulations ($\|\Phi\|_2\|\Phi^{-1}\|_2 \approx 5\text{e}6$). Il est à noter ici que le système filtre naturellement les hautes fréquences : la méthode d'exploration employée n'est peut-être pas la mieux adaptée.

Dans cet essai, le système considéré est simplement stable. Nous avons donc pu initialiser le correcteur K_0 à zéro. Dans le cas d'un système instable, il faut trouver un correcteur initial stabilisant le système ce qui n'est pas facile sans la connaissance du modèle du système. Une autre solution consisterait à changer



(a) Critère $V(X_0)$ en fonction des itérations (en pointillés la valeur optimale du critère)

(b) Position de l'objet en fonction du temps

FIGURE 6 – Résultats de simulation de l'algorithme 20 initialisé avec une politique nulle.

le protocole d'apprentissage. On pourrait par exemple recueillir des échantillons avant que le système ne diverge et utiliser ensuite ces échantillons pour l'apprentissage.

6 Résultats expérimentaux

6.1 Reconstruction de l'état

La caméra ne mesurant que la position de l'objet, il est nécessaire de reconstruire l'état du système en observant la commande et la sortie du système (position). Nous avons donc réalisé un observateur de Luenberger (Ellis, 2002). Cet observateur est de la forme :

$$\begin{cases} \hat{X}_{k+1} & = A\hat{X}_k + BU_k + L(Y_k - \hat{Y}_k) \\ \hat{Y}_k & = C\hat{X}_k \end{cases} \tag{21}$$

Nous avons choisi les coefficients de l'observateur par placement de pôles de façon à minimiser l'impact de l'observateur sur le critère $V(X_0)$. Les pôles de l'observateur sont placés à 0.8465, ce qui donne les coefficients $L = \begin{bmatrix} -0.0118 & 0.0047 & -0.0229 \end{bmatrix}'$.

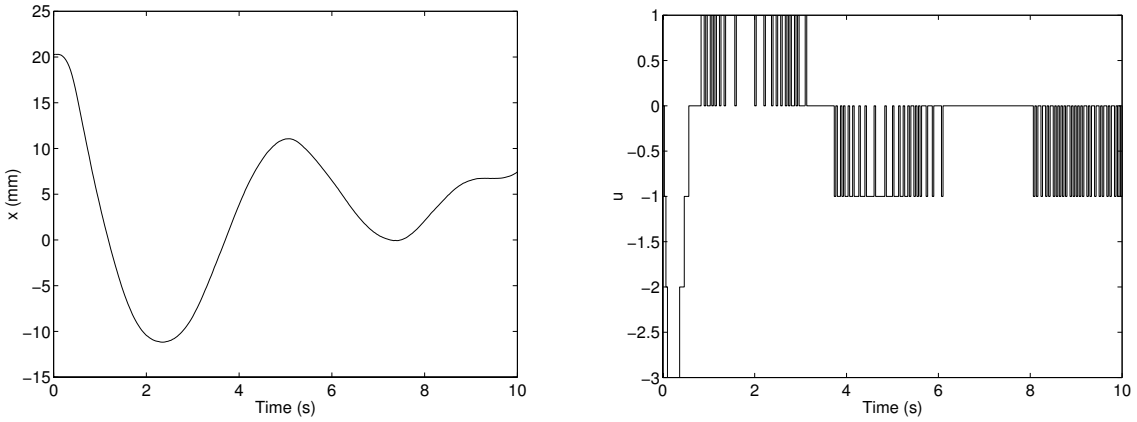
6.2 Commande linaire quadratique

Nous avons commencé par tester le correcteur linéaire quadratique optimal K^* calculé précédemment (cf. équation (20)). Les courbes de la figure 7 montrent les performances de ce correcteur sur le système réel.

On s'aperçoit qu'il y a une grande différence avec la simulation. En moyenne, le critère $V(X_0)$ obtenu est $2.3917\text{e}4$ soit quatre à cinq fois plus important qu'en simulation. La réponse du système réel est beaucoup moins amortie et le correcteur peine à ramener l'objet à zéro.

Cet écart peut s'expliquer par plusieurs facteurs :

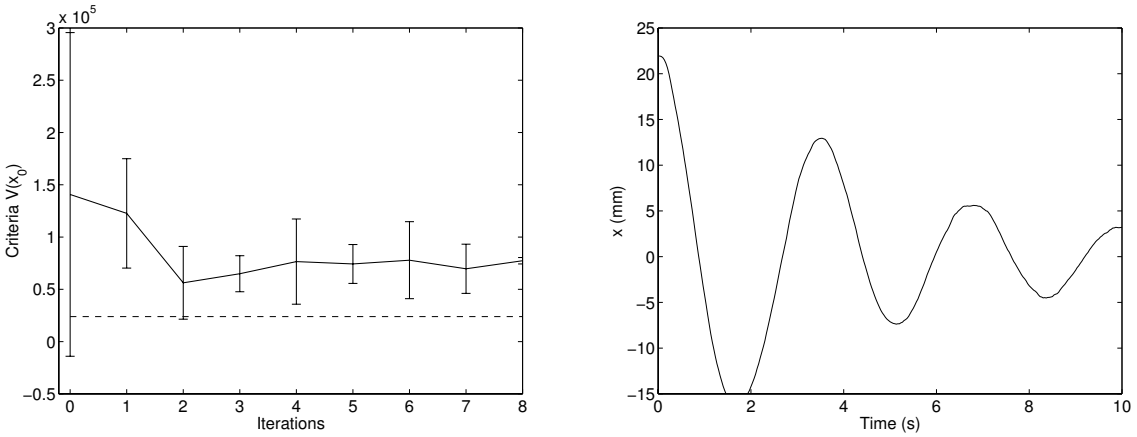
- une erreur d'identification des paramètres du modèle utilisé pour le calcul du correcteur optimal ;
- la présence de l'observateur ; en théorie, l'observateur doit avoir une dynamique bien supérieure aux constantes de temps du système pour ne pas influer sur la dynamique du système bouclé. Dans notre cas, la fréquence d'échantillonnage n'est pas tout à fait suffisante pour atteindre ce cas idéal ;
- certaines non-linéarités ne sont pas modélisées dans la simulation ; d'un part le système ne peut pas réaliser exactement les commandes souhaitées par le correcteur car les commandes exécutables sont des entiers et non des valeurs continues (quantification de la commande), d'autre part les commandes réelles sont bornées (saturation) ;
- le retard de la mesure de la position de l'objet (temps de transfert de l'image vidéo) ;



(a) Position de l'objet en fonction du temps

(b) Commande en fonction du temps

FIGURE 7 – Résultats expérimentaux avec le correcteur LQ.



(a) Critère $V(X_0)$ en fonction des itérations (moyenne sur 6 essais, en pointillés la valeur du critère pour la commande LQ)

(b) Position de l'objet en fonction du temps

FIGURE 8 – Résultats expérimentaux de l'algorithme 20 initialisé avec une politique nulle.

– les importantes perturbations subies par le système réel.

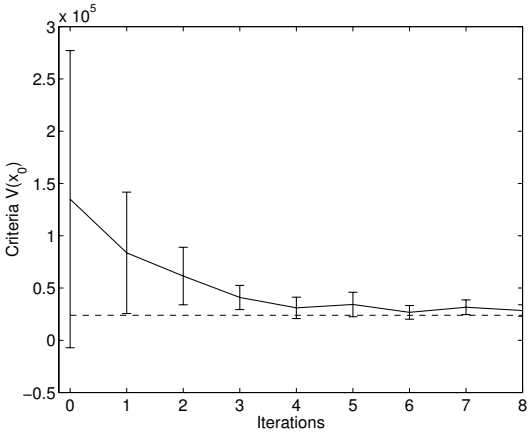
6.3 Apprentissage *ex nihilo*

Nous avons appliqué l'algorithme 20 au système réel en initialisant le correcteur à zéro comme en simulation. Cette application directe donne des résultats peu convaincants comme le montre la figure 8. Le critère est sensiblement amélioré mais semble osciller autour d'une valeur trois fois supérieure à celle de la commande LQ. La politique obtenue reste stabilisante mais les trajectoires obtenues sont encore moins amorties que celles de la commande LQ.

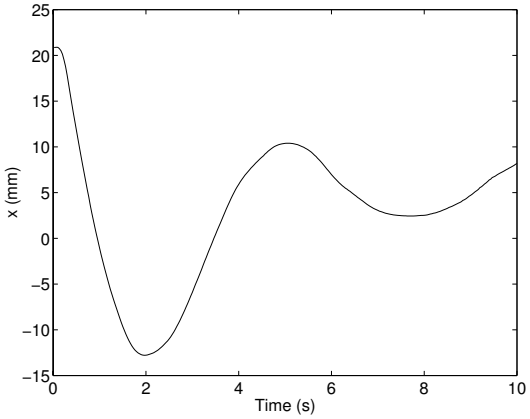
6.4 Fonction de formatage

Les conditions d'expérimentation étant identiques pour la commande LQ et pour l'apprentissage (mêmes contraintes sur les commandes, même observateur, même retard de mesure, etc.), on peut se demander pourquoi on obtient des performances inférieures dans le cas de l'apprentissage.

Le problème vient des contraintes sur les commandes. En effet, la commande LQ est fondée sur une identification paramétrique utilisant des échantillons obtenus à l'aide de commandes réalisables (par exemple $u = 2$). Or, la ligne 10 de l'algorithme 20 génère des commandes qui peuvent ne pas être réalisables. Il y a donc une erreur dans la plupart des échantillons entre la commande enregistrée et la commande véritable-



(a) Critère $V(X_0)$ en fonction des itérations (moyenne sur 7 essais, en pointillés la valeur du critère pour la commande LQ)



(b) Position de l'objet en fonction du temps

FIGURE 9 – Résultats expérimentaux de l'algorithme 20 avec fonction de formatage et initialisé avec une politique nulle.

ment réalisée. Cette erreur se retrouve dans le calcul de la fonction de valeur qui ne converge pas vers la fonction de valeur optimale pour le système commandé.

Nous avons donc remplacé la ligne 10 de l'algorithme 20 par la ligne suivante :

$$U_k \leftarrow f(K_t X_k + e_k) \tag{22}$$

f étant une fonction de formatage de la commande. Pour le système étudié ici, la fonction de formatage est définie par :

$$f(u) = \begin{cases} -5 & \text{si } u < -5 \\ 5 & \text{si } u > 5 \\ \text{Arrondi}(u) & \text{si } -5 \leq u \leq 5 \end{cases} \tag{23}$$

Le but est de générer des commandes U_k qui soient réalisables par le système. Ainsi, dans les échantillons utilisés pour l'apprentissage, les commandes enregistrées seront identiques aux commandes réalisées. *La fonction de formatage est propre au système et doit donc être définie par rapport aux contraintes techniques du dispositif expérimental.*

Nous avons testé l'algorithme utilisant la fonction de formatage. La figure 9 montre les résultats expérimentaux obtenus. Cette fois, l'apprentissage converge vers la même solution que la commande LQ. *Ces résultats montrent que l'on peut apprendre une commande optimale pour un système linéaire réel malgré ses imperfections et les perturbations du milieu ambiant*.

7 Conclusion

Dans cet article, nous avons présenté l'application de l'algorithme d'itération de la politique LSPI de Lagoudakis & Parr (2003) à la commande d'un système linéaire avec coût quadratique selon le protocole initialement proposé par Bradtke (1993). Cette méthode permet de trouver un correcteur optimal par retour d'état sans connaître le modèle du système (c'est-à-dire les matrices A, B, C et D). L'intérêt de cette approche est que le nombre de paramètres à déterminer est faible et que la politique gloutonne peut se déduire de la fonction de valeur de manière analytique. De plus, si la politique initiale est stabilisante la convergence vers une politique optimale est garantie en un nombre fini d'itérations.

Les simulations sur un système d'ordre trois ont montré que la convergence de l'algorithme est particulièrement rapide (quelques itérations) et que le correcteur obtenu est bien optimal. Les résultats expérimentaux

sur une surface de manipulation à coussin d'air ont mis en évidence la nécessité de formater les commandes générées par l'algorithme. Ce formatage a pour objectif d'éviter la génération de commandes irréalisables qui introduisent un biais dans la mise à jour de la fonction de valeur. L'apprentissage converge alors vers la même solution que la commande linéaire quadratique.

Une première perspective d'amélioration vise à traiter le problème de la reconstruction de l'état. Dans les expériences réalisées, la caméra ne mesurant que la position, l'état du système est reconstruit à l'aide d'un observateur de Luenberger. Cet observateur nécessite la connaissance du modèle du système (matrices A, B, C et D). A l'avenir, il serait intéressant d'étendre l'algorithme utilisé au cas partiellement observable pour se passer de ce genre d'observateur. L'ambition finale est d'obtenir une méthode de commande optimale adaptative ne nécessitant pas la connaissance du modèle ni pour la reconstruction d'état ni pour la synthèse du correcteur.

Une autre perspective concerne le signal d'exploration. Nous avons rencontré des problèmes de stabilité numérique dans le cas où l'exploration n'était pas suffisante. Augmenter simplement le niveau de bruit n'est possible que dans une certaine mesure car cela finit par déstabiliser le système. Il serait donc intéressant de concevoir un signal respectant la condition de Goodwin and Sin (Bradtke *et al.*, 1994) tout en garantissant la stabilité du système. Ceci plaide en faveur d'un apprentissage actif de la loi de commande.

Références

BERTSEKAS D. P. (2005). *Dynamic Programming and Optimal Control, Volume 1*. Athena Scientific.

BOYAN J. A. (2002). Technical update : Least-squares temporal difference learning. *Machine Learning*, **49**, 233–246.

BRADTKE S. J. (1993). Reinforcement learning applied to linear quadratic regulation. In *Proc. of Advances in Neural Information Processing Systems*, p. 295–302 : Morgan Kaufmann.

BRADTKE S. J. & BARTO A. G. (1996). Linear least-squares algorithms for temporal difference learning. *Machine Learning*, **22**, 33–57.

BRADTKE S. J., YDSTIE B. E. & BARTO A. G. (1994). Adaptive linear quadratic control using policy iteration. In *Proc. of the American Control Conference*, p. 3475–3476, Baltimore, Maryland.

ELLIS G. (2002). *Observers in Control Systems : A Practical Guide*. Academic Press, Elsevier Science.

GERAMIFARD A., BOWLING M. & SUTTON R. S. (2006). Incremental least-squares temporal difference learning. In *Proc. of the National Conference on Artificial Intelligence*, p. 356–361.

LAGOUDAKIS M. G. & PARR R. (2003). Least-squares policy iteration. *Journal of Machine Learning Research*, **4**, 1107–1149.

PETERS J. & SCHAAL S. (2008). Natural actor-critic. *Neurocomputing*, **71**, 1180–1190.

WALTHAM C., BENDALL S. & KOTLICKI A. (2003). Bernoulli levitation. *American Journal of Physics*, **71**(2), 176–179.